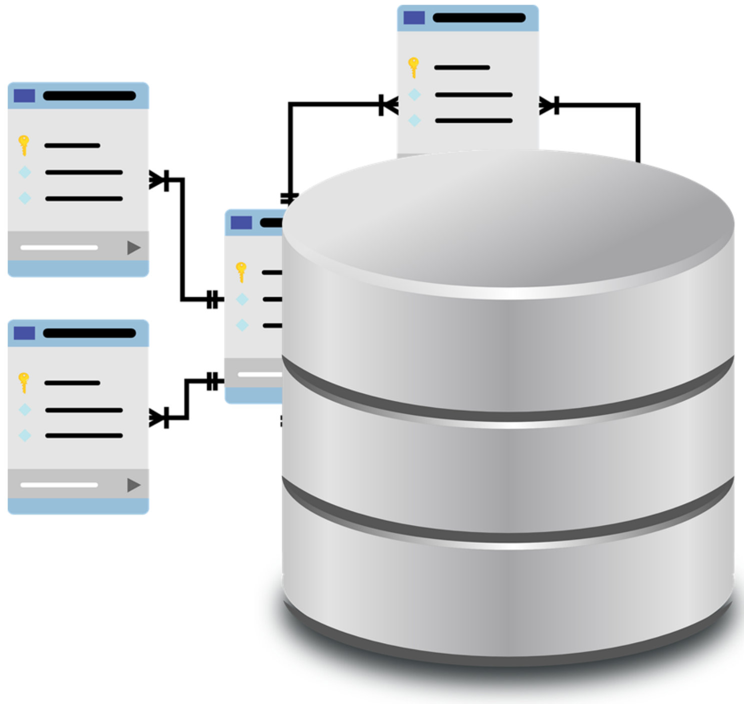
SQL Data Definition Language



Create and Insert

SQL Create and Use Database

Let's create a database named “mystore”

Item_no	Item_name	Unit_Price	Inventory
2321	Dell Laptop	1500	56
5432	Seagate Drive	200	100
5674	Kingston USB Drive	70	500
8542	Backpack	100	45

Introduction to Data Science: SQL

```
-- Create database for mystore
CREATE SCHEMA mystore;

-- Create table named inventory in mystore
CREATE TABLE mystore.inventory(Item_no INT NOT
NULL,  Item_name VARCHAR(100) NOT NULL,
Unit_Price INT NOT NULL, Inventory INT,  PRIMARY KEY
(Item_no));

-- Populate table with values/data
INSERT INTO mystore.inventory (Item_no, Item_name,
Unit_Price, Inventory) VALUES (2321, 'Dell Laptop',
1500, 56);
INSERT INTO mystore.inventory (Item_no, Item_name,
Unit_Price, Inventory) VALUES (5432, 'Seagate Drive',
200, 100);
INSERT INTO mystore.inventory (Item_no, Item_name,
Unit_Price, Inventory) VALUES (5674, 'Kingston USB
Drive', 70, 500);
INSERT INTO mystore.inventory (Item_no, Item_name,
Unit_Price, Inventory) VALUES (8542, 'Backpack', 100,
45);
```

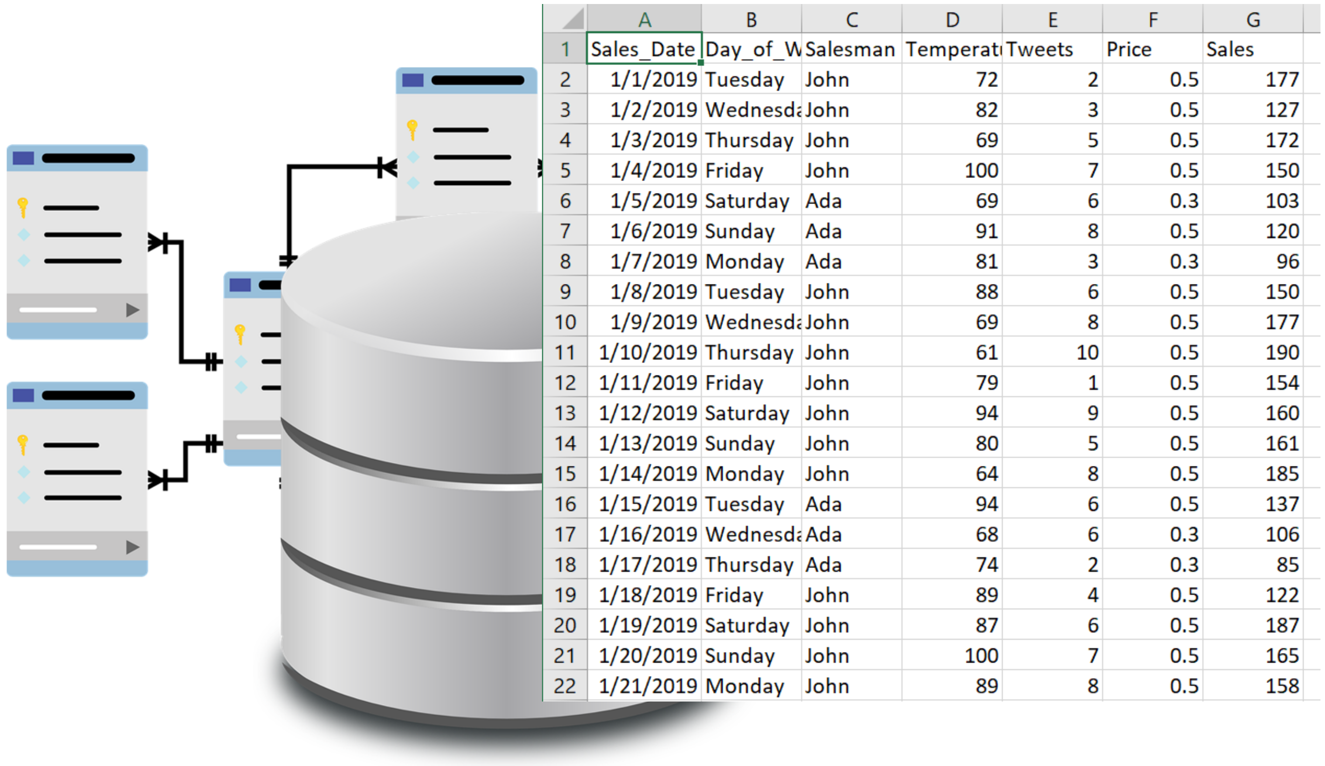
Item_no	Item_name	Unit_Price	Inventory
2321	Dell Laptop	1500	56
5432	Seagate Drive	200	100
5674	Kingston USB Drive	70	500
8542	Backpack	100	45

Introduction to Data Science: SQL DDL

```
-- Select database to use
USE mystore;

/* once USE mystore is executed, we can eliminate
   the dot operator and database name */
CREATE TABLE inventory(Item_no INT NOT NULL,
Item_name VARCHAR(100) NOT NULL,  Unit_Price INT
NOT NULL, Inventory INT,  PRIMARY KEY (Item_no));

-- Insert new product with null value
INSERT INTO inventory (Item_no, Item_name,
Unit_Price, Inventory) VALUES (2348, ‘HP Laptop', 1000,
null);

-- Insert new product with no null columns only
INSERT INTO inventory (Item_no, Item_name,
Unit_Price) VALUES (7344, ‘Lenovo Laptop', 988);
```

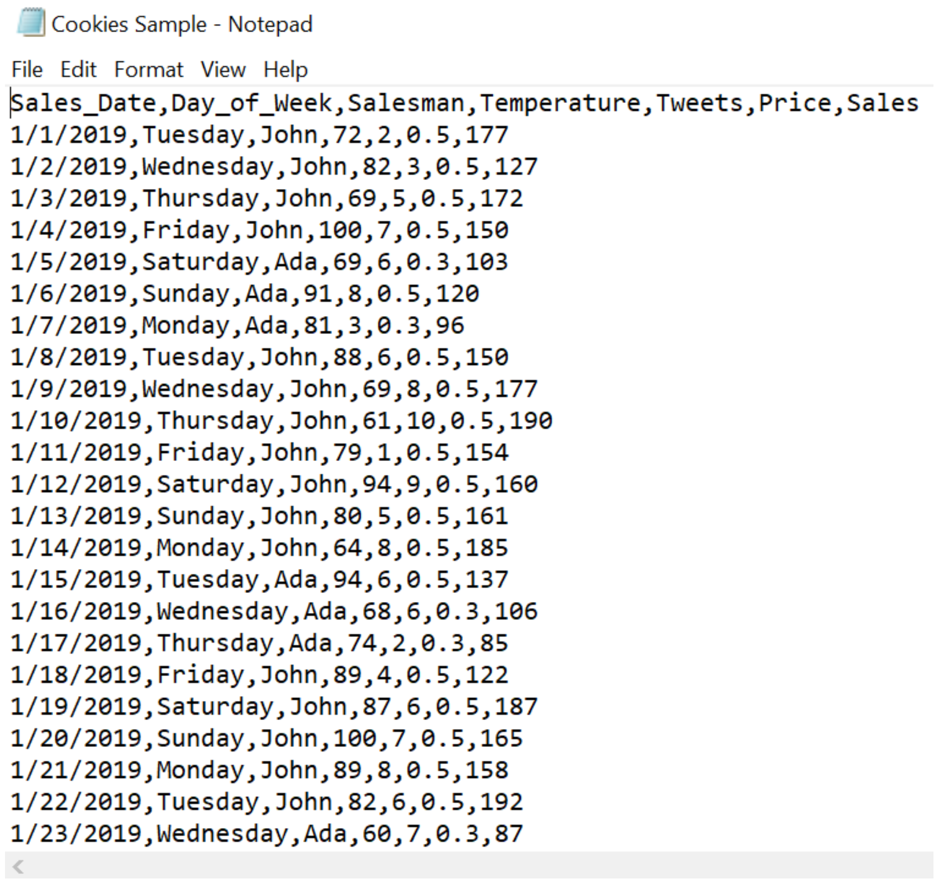
Item_no	Item_name	Unit_Price	Inventory
2321	Dell Laptop	1500	56
5432	Seagate Drive	200	100
5674	Kingston USB Drive	70	500
8542	Backpack	100	45

SQL Load Data from CSV File



	A	B	C	D	E	F	G
1	Sales_Date	Day_of_W	Salesman	Temperat	Tweets	Price	Sales
2	1/1/2019	Tuesday	John	72	2	0.5	177
3	1/2/2019	Wednesd	John	82	3	0.5	127
4	1/3/2019	Thursday	John	69	5	0.5	172
5	1/4/2019	Friday	John	100	7	0.5	150
6	1/5/2019	Saturday	Ada	69	6	0.3	103
7	1/6/2019	Sunday	Ada	91	8	0.5	120
8	1/7/2019	Monday	Ada	81	3	0.3	96
9	1/8/2019	Tuesday	John	88	6	0.5	150
10	1/9/2019	Wednesd	John	69	8	0.5	177
11	1/10/2019	Thursday	John	61	10	0.5	190
12	1/11/2019	Friday	John	79	1	0.5	154
13	1/12/2019	Saturday	John	94	9	0.5	160
14	1/13/2019	Sunday	John	80	5	0.5	161
15	1/14/2019	Monday	John	64	8	0.5	185
16	1/15/2019	Tuesday	Ada	94	6	0.5	137
17	1/16/2019	Wednesd	Ada	68	6	0.3	106
18	1/17/2019	Thursday	Ada	74	2	0.3	85
19	1/18/2019	Friday	John	89	4	0.5	122
20	1/19/2019	Saturday	John	87	6	0.5	187
21	1/20/2019	Sunday	John	100	7	0.5	165
22	1/21/2019	Monday	John	89	8	0.5	158

SQL Load Data from CSV File



```
Sales_Date,Day_of_Week,Salesman,Temperature,Tweets,Price,Sales
1/1/2019,Tuesday,John,72,2,0.5,177
1/2/2019,Wednesday,John,82,3,0.5,127
1/3/2019,Thursday,John,69,5,0.5,172
1/4/2019,Friday,John,100,7,0.5,150
1/5/2019,Saturday,Ada,69,6,0.3,103
1/6/2019,Sunday,Ada,91,8,0.5,120
1/7/2019,Monday,Ada,81,3,0.3,96
1/8/2019,Tuesday,John,88,6,0.5,150
1/9/2019,Wednesday,John,69,8,0.5,177
1/10/2019,Thursday,John,61,10,0.5,190
1/11/2019,Friday,John,79,1,0.5,154
1/12/2019,Saturday,John,94,9,0.5,160
1/13/2019,Sunday,John,80,5,0.5,161
1/14/2019,Monday,John,64,8,0.5,185
1/15/2019,Tuesday,Ada,94,6,0.5,137
1/16/2019,Wednesday,Ada,68,6,0.3,106
1/17/2019,Thursday,Ada,74,2,0.3,85
1/18/2019,Friday,John,89,4,0.5,122
1/19/2019,Saturday,John,87,6,0.5,187
1/20/2019,Sunday,John,100,7,0.5,165
1/21/2019,Monday,John,89,8,0.5,158
1/22/2019,Tuesday,John,82,6,0.5,192
1/23/2019,Wednesday,Ada,60,7,0.3,87
```

SQL Load Data from CSV File

Step 1: Understand the structure (column heading, data type)

Step 2: Create SQL script

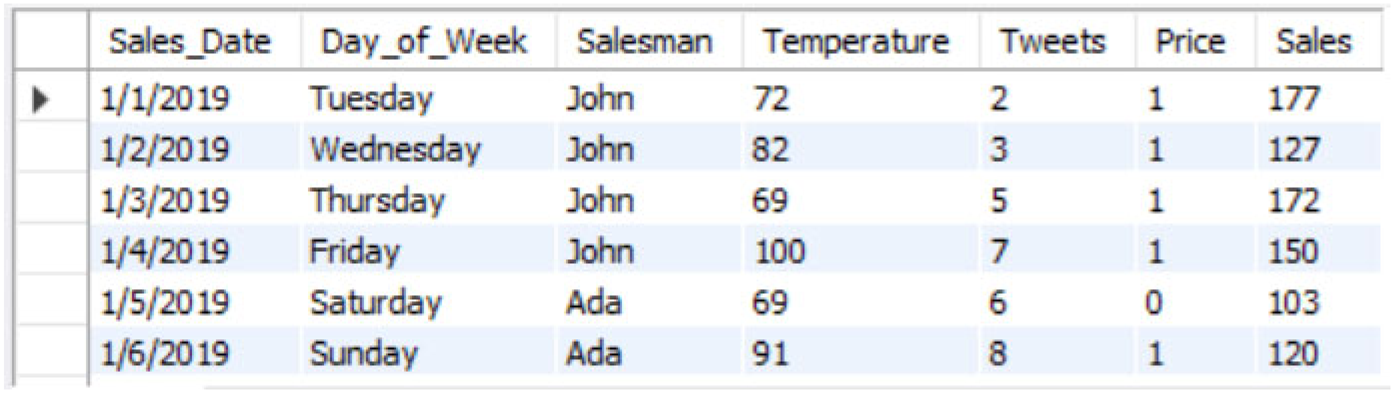
- Create database
- Create table with the same structure observed from CSV file
- Load data from CSV file

Step 3: Run SQL script

SQL Load Data from CSV File

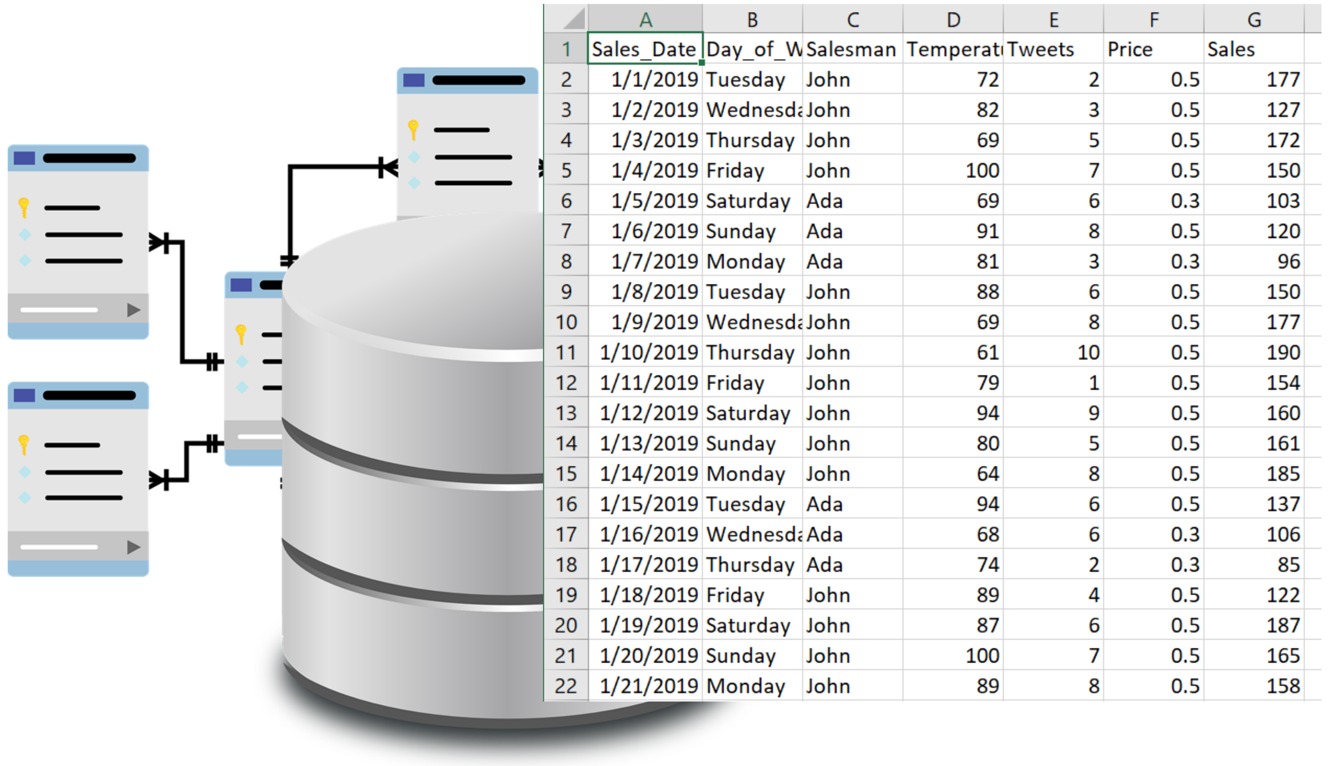
```
CREATE SCHEMA cookies;

CREATE TABLE cookies.sales
(Sales_Date varchar(10),
Day_of_Week varchar(10),
Salesman varchar(10),
Temperature INT,
Tweets INT,
Price INT,
Sales INT);

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Cookies Sample.csv'
INTO TABLE cookies.sales
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n‘
IGNORE 1 ROWS;
```

SQL Load Data from CSV File



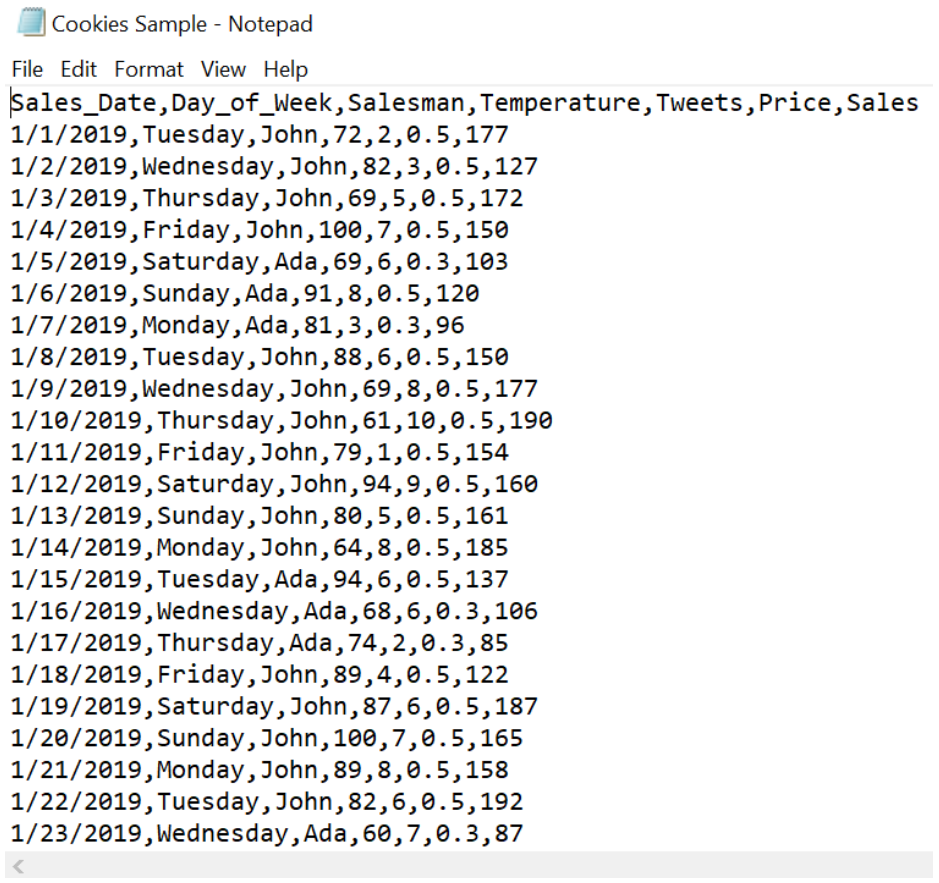
Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
1/1/2019	Tuesday	John	72	2	1	177
1/2/2019	Wednesday	John	82	3	1	127
1/3/2019	Thursday	John	69	5	1	172
1/4/2019	Friday	John	100	7	1	150
1/5/2019	Saturday	Ada	69	6	0	103
1/6/2019	Sunday	Ada	91	8	1	120

Python MySQL Load Data from CSV File



	A	B	C	D	E	F	G
1	Sales_Date	Day_of_W	Salesman	Temperat	Tweets	Price	Sales
2	1/1/2019	Tuesday	John	72	2	0.5	177
3	1/2/2019	Wednesda	John	82	3	0.5	127
4	1/3/2019	Thursday	John	69	5	0.5	172
5	1/4/2019	Friday	John	100	7	0.5	150
6	1/5/2019	Saturday	Ada	69	6	0.3	103
7	1/6/2019	Sunday	Ada	91	8	0.5	120
8	1/7/2019	Monday	Ada	81	3	0.3	96
9	1/8/2019	Tuesday	John	88	6	0.5	150
10	1/9/2019	Wednesda	John	69	8	0.5	177
11	1/10/2019	Thursday	John	61	10	0.5	190
12	1/11/2019	Friday	John	79	1	0.5	154
13	1/12/2019	Saturday	John	94	9	0.5	160
14	1/13/2019	Sunday	John	80	5	0.5	161
15	1/14/2019	Monday	John	64	8	0.5	185
16	1/15/2019	Tuesday	Ada	94	6	0.5	137
17	1/16/2019	Wednesda	Ada	68	6	0.3	106
18	1/17/2019	Thursday	Ada	74	2	0.3	85
19	1/18/2019	Friday	John	89	4	0.5	122
20	1/19/2019	Saturday	John	87	6	0.5	187
21	1/20/2019	Sunday	John	100	7	0.5	165
22	1/21/2019	Monday	John	89	8	0.5	158

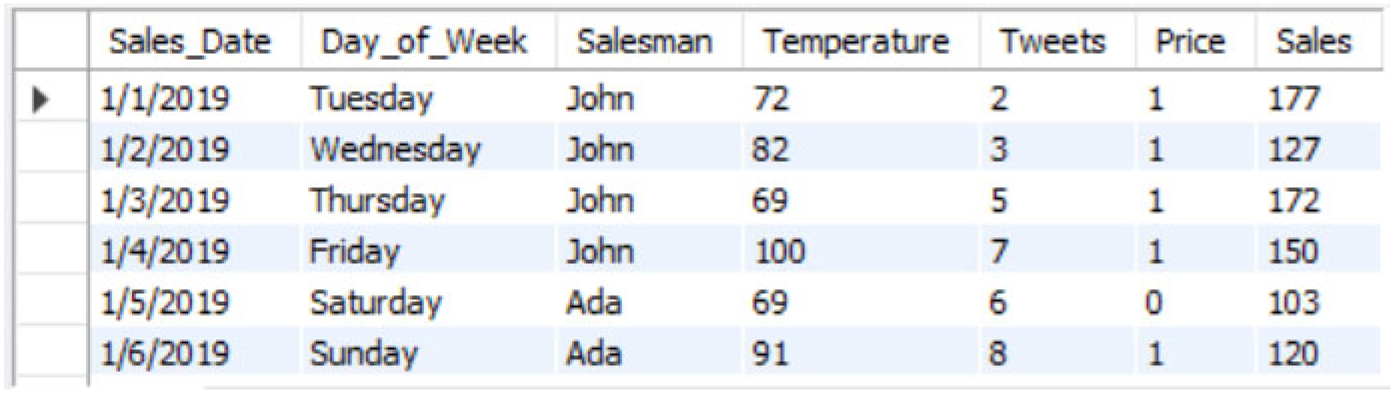
Python MySQL Load Data from CSV File



```
Sales_Date,Day_of_Week,Salesman,Temperature,Tweets,Price,Sales
1/1/2019,Tuesday,John,72,2,0.5,177
1/2/2019,Wednesday,John,82,3,0.5,127
1/3/2019,Thursday,John,69,5,0.5,172
1/4/2019,Friday,John,100,7,0.5,150
1/5/2019,Saturday,Ada,69,6,0.3,103
1/6/2019,Sunday,Ada,91,8,0.5,120
1/7/2019,Monday,Ada,81,3,0.3,96
1/8/2019,Tuesday,John,88,6,0.5,150
1/9/2019,Wednesday,John,69,8,0.5,177
1/10/2019,Thursday,John,61,10,0.5,190
1/11/2019,Friday,John,79,1,0.5,154
1/12/2019,Saturday,John,94,9,0.5,160
1/13/2019,Sunday,John,80,5,0.5,161
1/14/2019,Monday,John,64,8,0.5,185
1/15/2019,Tuesday,Ada,94,6,0.5,137
1/16/2019,Wednesday,Ada,68,6,0.3,106
1/17/2019,Thursday,Ada,74,2,0.3,85
1/18/2019,Friday,John,89,4,0.5,122
1/19/2019,Saturday,John,87,6,0.5,187
1/20/2019,Sunday,John,100,7,0.5,165
1/21/2019,Monday,John,89,8,0.5,158
1/22/2019,Tuesday,John,82,6,0.5,192
1/23/2019,Wednesday,Ada,60,7,0.3,87
```

```
import mysql.connector as sq
mydb=sq.connect(host="localhost",user="root",passwd="ucla", buffered=True)

mycursor = mydb.cursor()
mycursor.execute('CREATE SCHEMA cookies')

SQLCMD = 'CREATE TABLE cookies.sales (Sales_Date varchar(10), \
Day_of_Week varchar(10), Salesman varchar(10), Temperature INT, \
Tweets INT, Price FLOAT, Sales INT)'

mycursor.execute(SQLCMD)

SQLCMD = "LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Cookies Sample.csv' \
INTO TABLE cookies.sales FIELDS TERMINATED BY ',' LINES TERMINATED BY '\\n' IGNORE 1 ROWS"

mycursor.execute(SQLCMD)
mydb.commit()
```

Python MySQL Load Data from CSV File

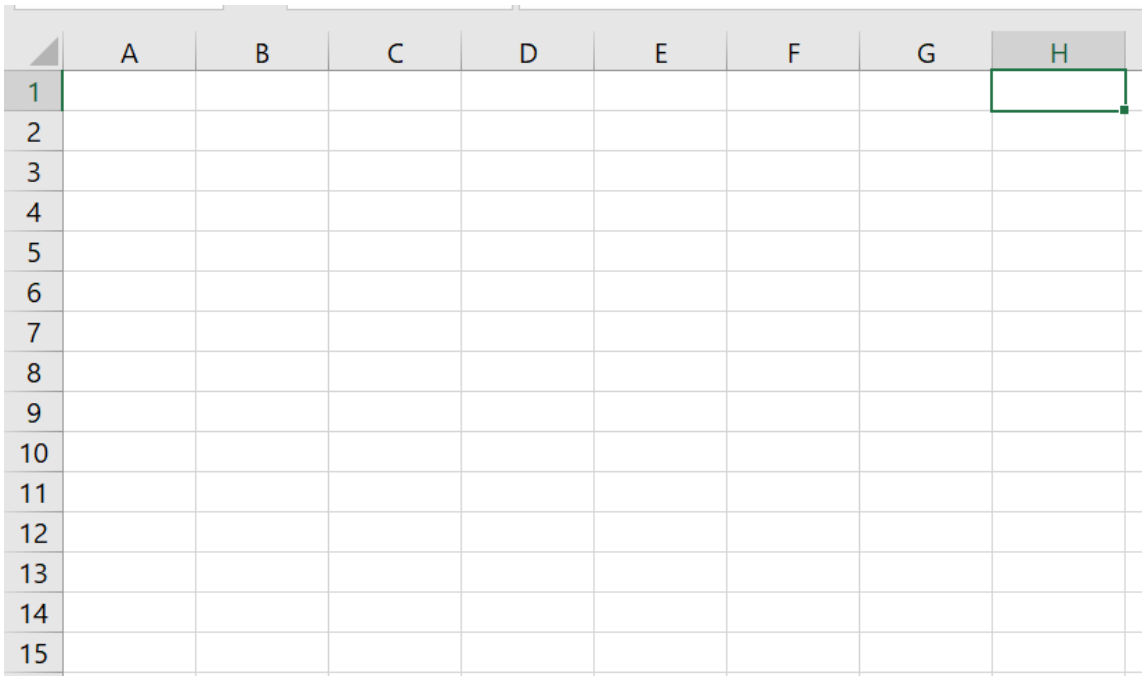


	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	1/1/2019	Tuesday	John	72	2	1	177
	1/2/2019	Wednesday	John	82	3	1	127
	1/3/2019	Thursday	John	69	5	1	172
	1/4/2019	Friday	John	100	7	1	150
	1/5/2019	Saturday	Ada	69	6	0	103
	1/6/2019	Sunday	Ada	91	8	1	120

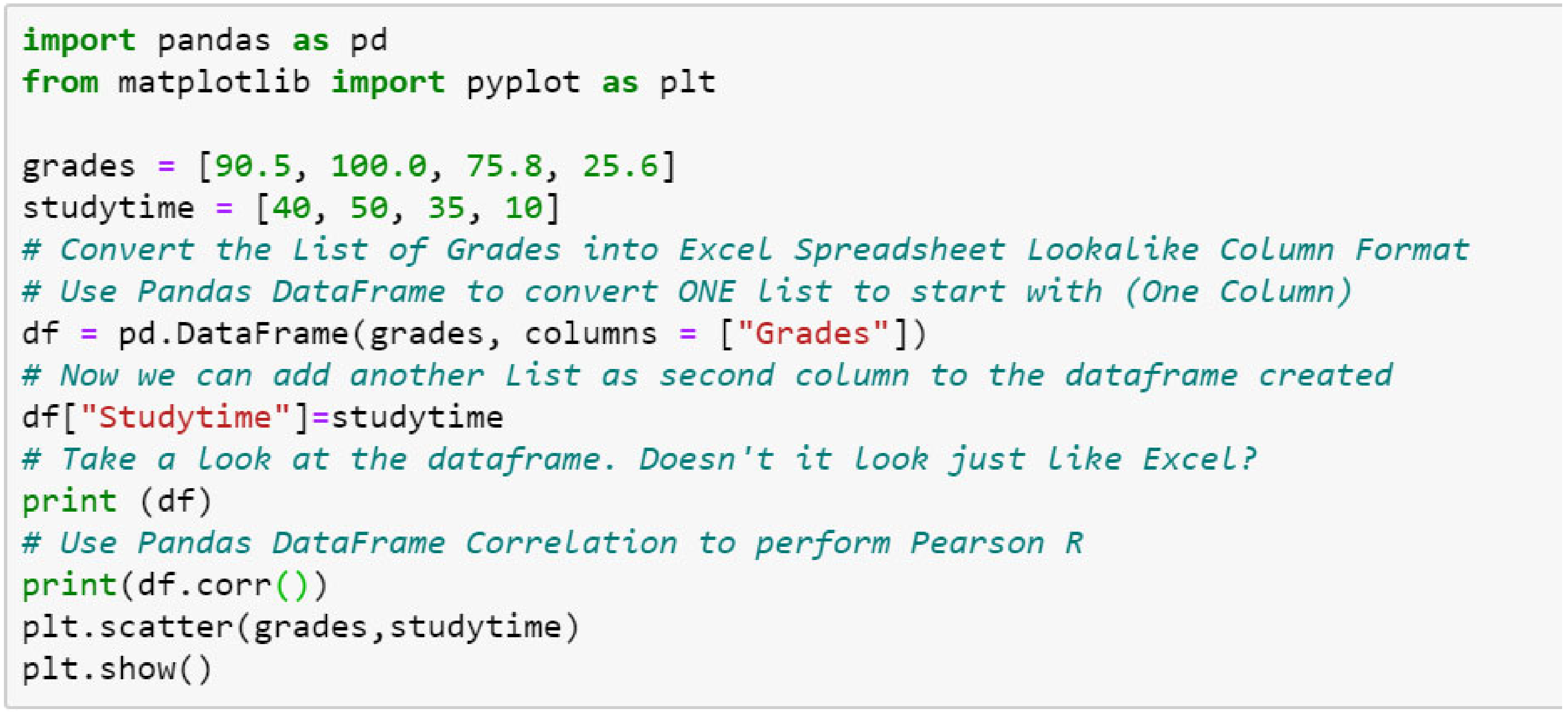
Python Pandas DataFrame

DataFrame is a 2-dimensional labeled data structure with columns of potentially different types

Data Science: Python Pandas DataFrame



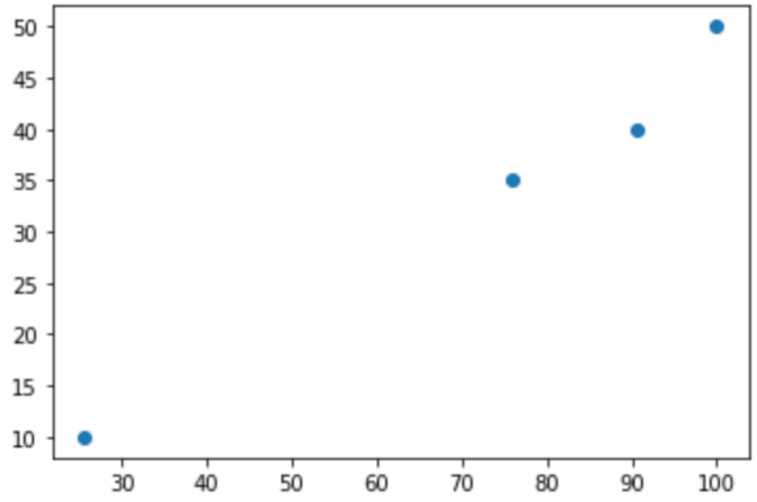
Python Pandas DataFrame



```python
import pandas as pd
from matplotlib import pyplot as plt

grades = [90.5, 100.0, 75.8, 25.6]
studytime = [40, 50, 35, 10]
# Convert the List of Grades into Excel Spreadsheet Lookalike Column Format
# Use Pandas DataFrame to convert ONE list to start with (One Column)
df = pd.DataFrame(grades, columns = ["Grades"])
# Now we can add another List as second column to the dataframe created
df["Studytime"]=studytime
# Take a look at the dataframe. Doesn't it look just like Excel?
print (df)
# Use Pandas DataFrame Correlation to perform Pearson R
print(df.corr())
plt.scatter(grades,studytime)
plt.show()
```

Data Science: Python Pandas DataFrame and Correlation

```
   Grades  Studytime
0    90.5         40
1   100.0         50
2    75.8         35
3    25.6         10
            Grades  Studytime
Grades     1.00000    0.99219
Studytime  0.99219    1.00000
```



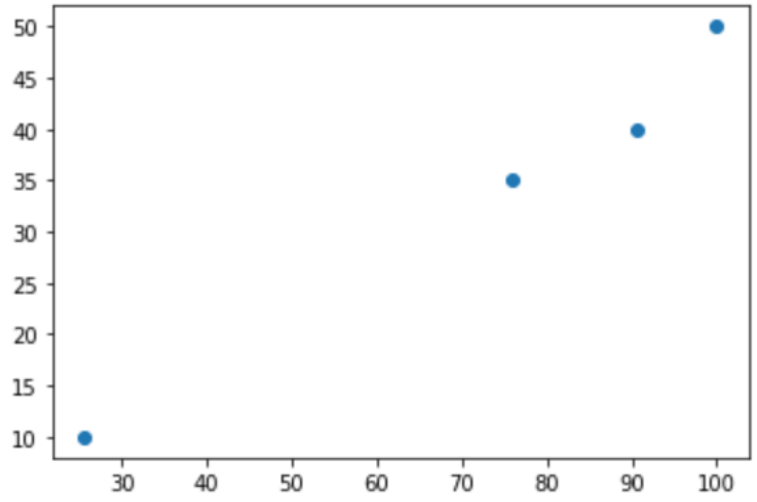
Data Science: Python Pandas DataFrame and Correlation Demo

Python Pandas DataFrame

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline

grades = [90.5, 100.0, 75.8, 25.6]
studytime = [40, 50, 35, 10]

# Convert the List of Grades into Excel Spreadsheet Lookalike Column Format
data = list(zip(grades, studytime))

df = pd.DataFrame(data, columns = ["Grades", "StudyTime"])

print (df)

# Use Pandas DataFrame Correlation
print (df.corr())
plt.scatter(studytime, grades)
plt.show()
```

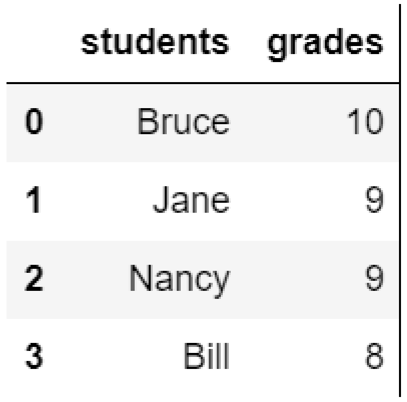
Data Science: Python Pandas DataFrame and Correlation

```
   Grades  Studytime
0    90.5         40
1   100.0         50
2    75.8         35
3    25.6         10
             Grades  Studytime
Grades      1.00000    0.99219
Studytime  0.99219    1.00000
```



Python Pandas DataFrame

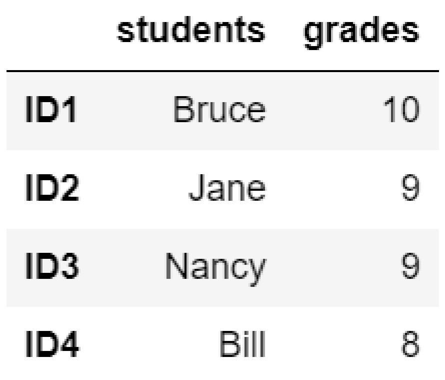
```
import pandas as pd

MyClass = {'students':['Bruce', 'Jane', 'Nancy',
'Bill'],
       'grades':[10, 9, 9, 8]}

df = pd.DataFrame(MyClass)
```



	students	grades
0	Bruce	10
1	Jane	9
2	Nancy	9
3	Bill	8

Python Pandas DataFrame

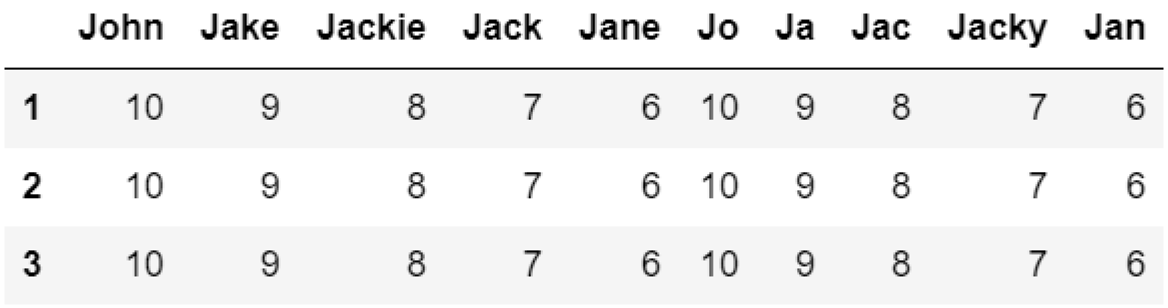
```
import pandas as pd

MyClass = {'students':['Bruce', 'Jane', 'Nancy',
'Bill'],
      'grades':[10, 9, 9, 8]}

df = pd.DataFrame(MyClass, index
=["ID1","ID2","ID3","ID4"])
```



	students	grades
ID1	Bruce	10
ID2	Jane	9
ID3	Nancy	9
ID4	Bill	8

Python Pandas DataFrame

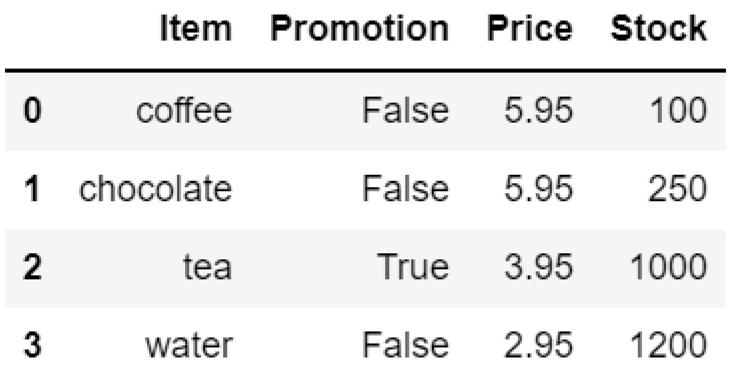
```
import pandas as pd

MyClass =
{'John':10,'Jake':9,'Jackie':8,'Jack':7,'Jane':6,'Jo'
:10,'Ja':9,'Jac':8,'Jacky':7,'Jan':6}

df = pd.DataFrame(MyClass, index=[1, 2, 3])
```



	John	Jake	Jackie	Jack	Jane	Jo	Ja	Jac	Jacky	Jan
1	10	9	8	7	6	10	9	8	7	6
2	10	9	8	7	6	10	9	8	7	6
3	10	9	8	7	6	10	9	8	7	6

Python Pandas DataFrame

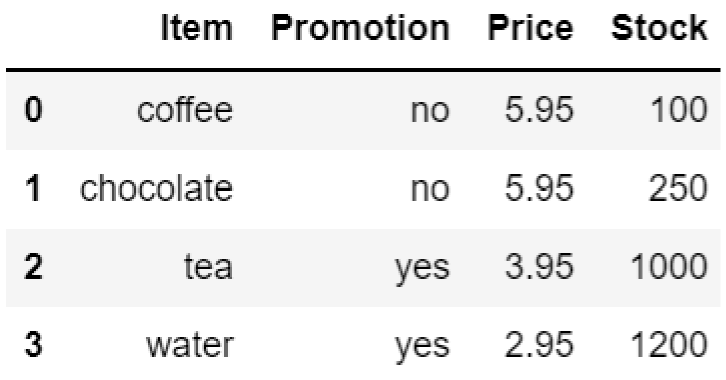
```
MyInventory = {
  "Item": ["coffee", "chocolate", "tea",
"water"],
  "Promotion": [False, False, True, False],
  "Price": [5.95, 5.95, 3.95, 2.95],
  "Stock": [100, 250, 1000, 1200]
}

ddf = pd.DataFrame(MyInventory)
ddf
```



	Item	Promotion	Price	Stock
0	coffee	False	5.95	100
1	chocolate	False	5.95	250
2	tea	True	3.95	1000
3	water	False	2.95	1200

Python Pandas DataFrame

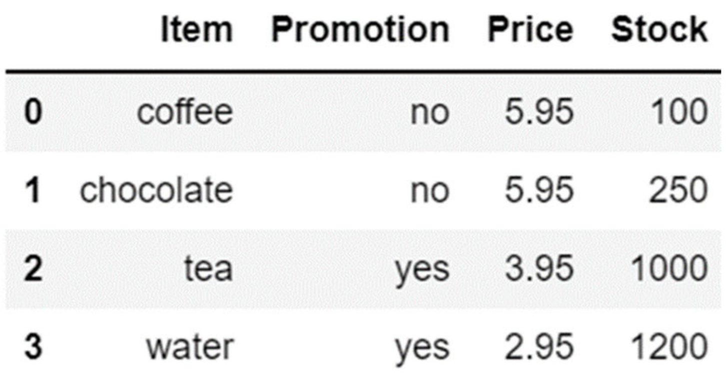
```
Inv2 = {
  "Item": ["coffee", "chocolate", "tea",
"water"],
  "Promotion": ["no","no","yes","yes"],
  "Price": [5.95, 5.95, 3.95, 2.95],
  "Stock": [100, 250, 1000, 1200]
}

InvDF = pd.DataFrame(Inv2)
InvDF
```



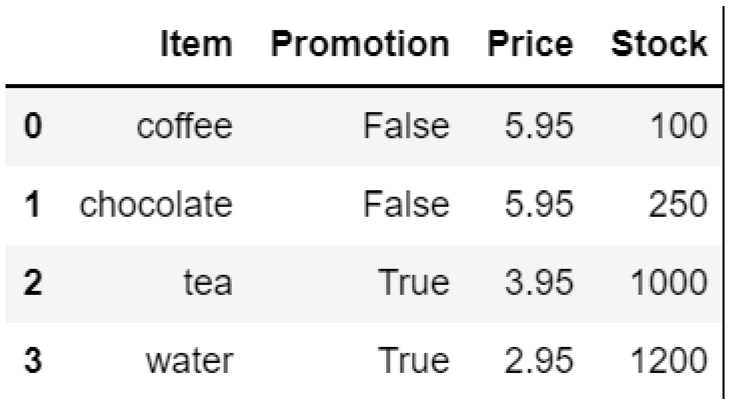
	Item	Promotion	Price	Stock
0	coffee	no	5.95	100
1	chocolate	no	5.95	250
2	tea	yes	3.95	1000
3	water	yes	2.95	1200

Python Pandas DataFrame

```
Inv2 = {
  "Item": ["coffee", "chocolate", "tea",
"water"],
  "Promotion": ["no","no","yes","yes"],
  "Price": [5.95, 5.95, 3.95, 2.95],
  "Stock": [100, 250, 1000, 1200]
}

InvDF = pd.DataFrame(Inv2)
InvDF

InvDF = InvDF.replace({'Promotion': {'no':
False,'yes': True}})
```



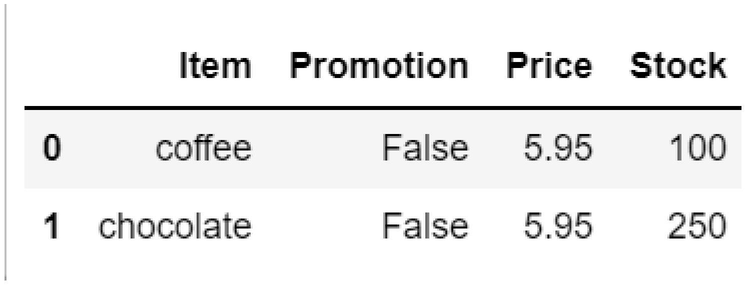
	Item	Promotion	Price	Stock
0	coffee	no	5.95	100
1	chocolate	no	5.95	250
2	tea	yes	3.95	1000
3	water	yes	2.95	1200



	Item	Promotion	Price	Stock
0	coffee	False	5.95	100
1	chocolate	False	5.95	250
2	tea	True	3.95	1000
3	water	True	2.95	1200

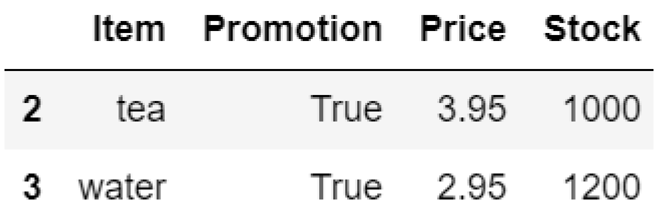
Python Pandas DataFrame

```
InvDF[InvDF["Promotion"] == False]
```

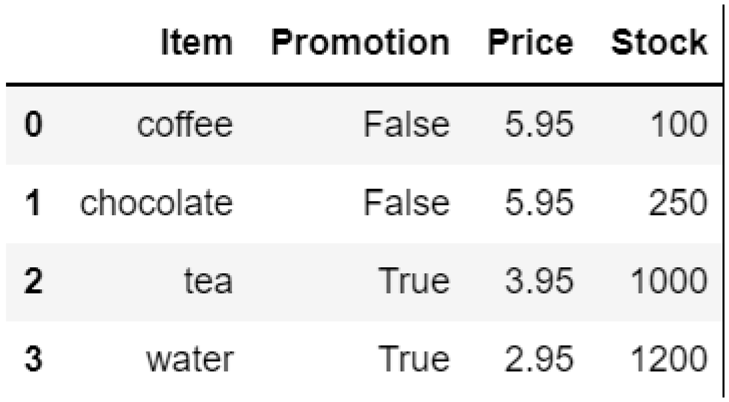


	Item	Promotion	Price	Stock
0	coffee	False	5.95	100
1	chocolate	False	5.95	250

```
InvDF[InvDF["Price"] < 5]
```

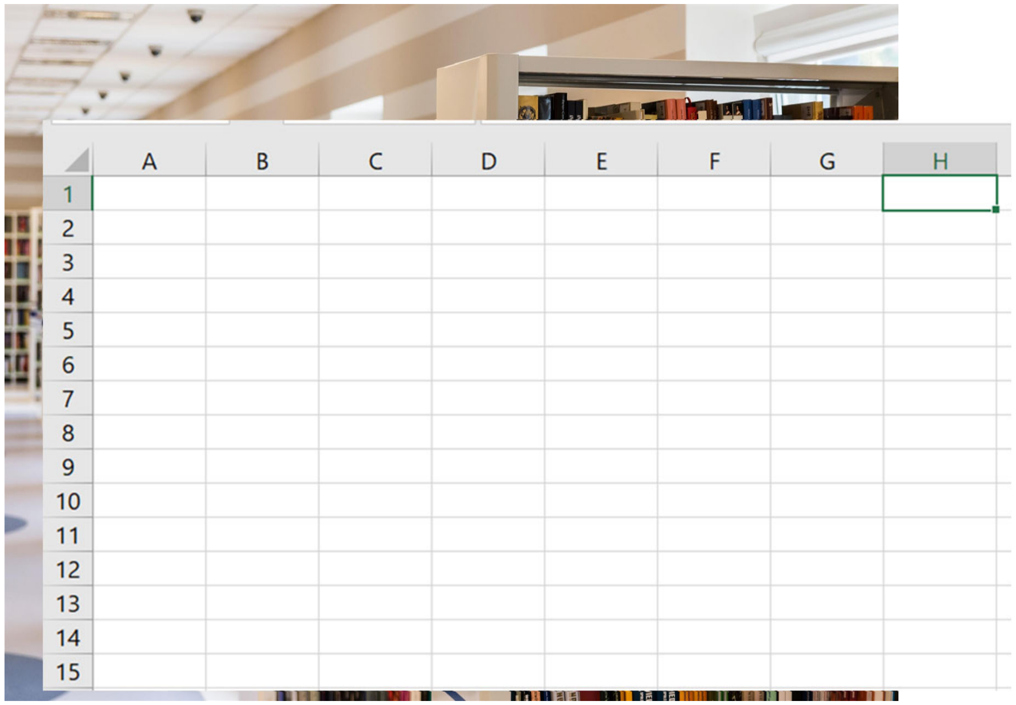


	Item	Promotion	Price	Stock
2	tea	True	3.95	1000
3	water	True	2.95	1200



	Item	Promotion	Price	Stock
0	coffee	False	5.95	100
1	chocolate	False	5.95	250
2	tea	True	3.95	1000
3	water	True	2.95	1200

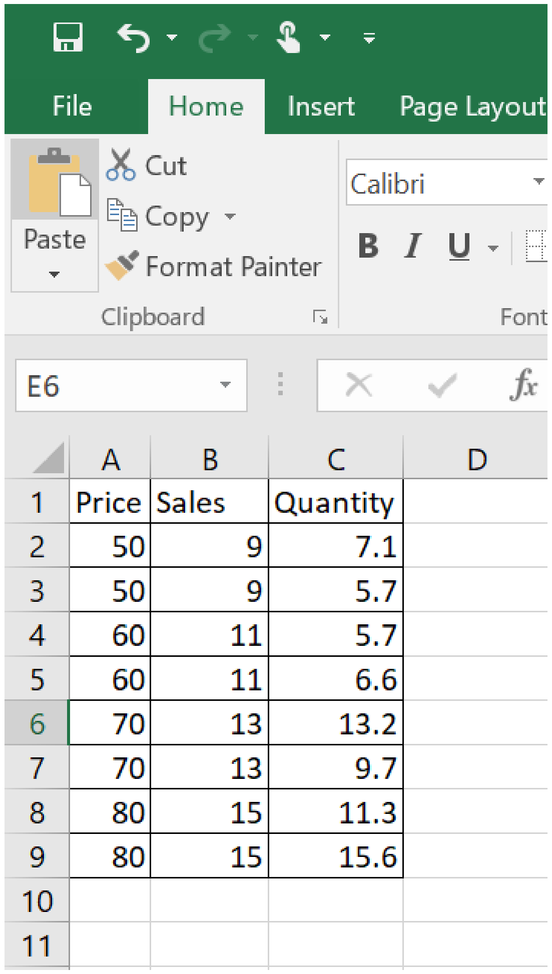
Data Science: Python Pandas DataFrame from Excel



Pandas **DataFrame** Review

DataFrame is a 2-dimensional labeled data structure with columns of potentially different types

Data Science: Python Pandas DataFrame from Excel



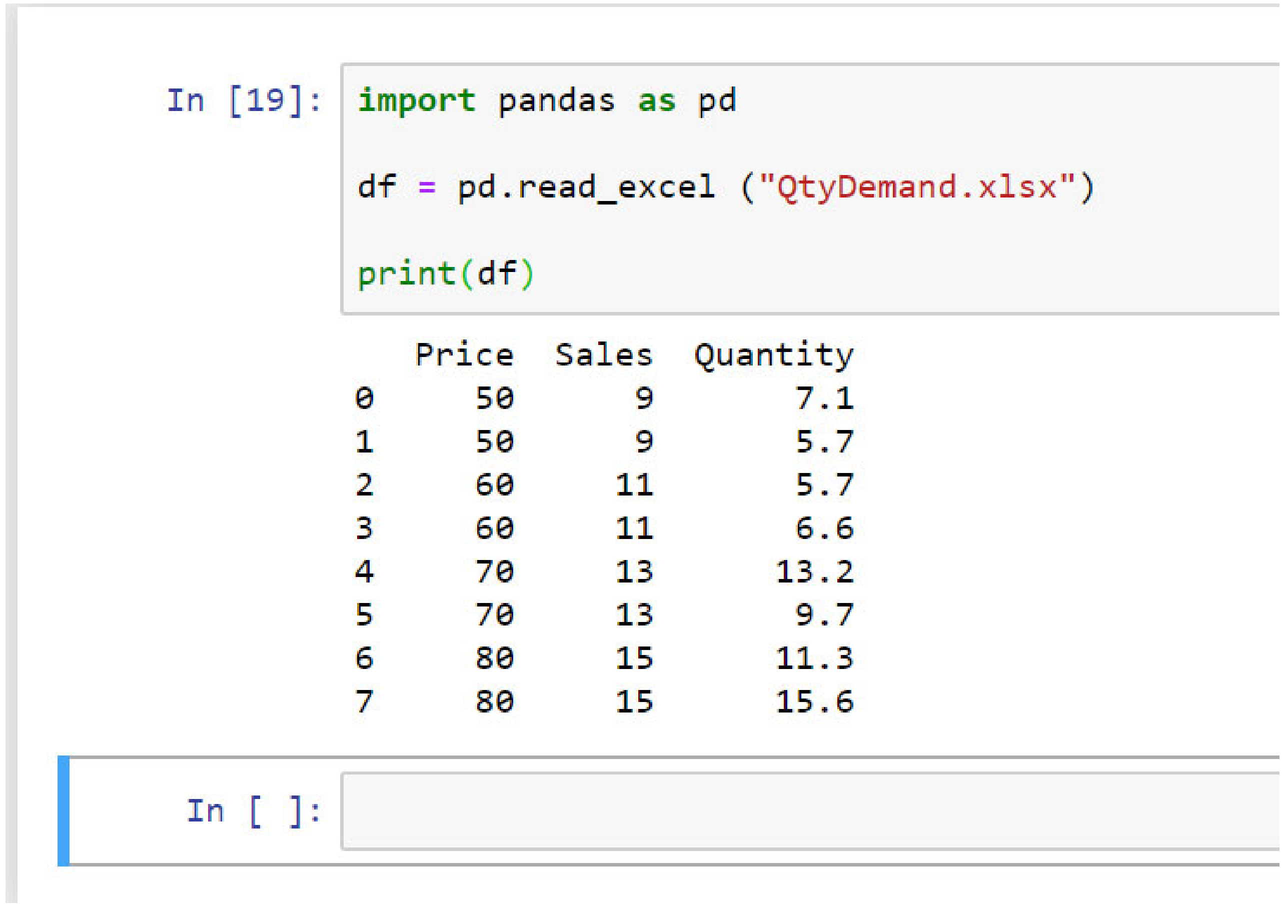
	A	B	C	D
1	Price	Sales	Quantity	
2	50	9	7.1	
3	50	9	5.7	
4	60	11	5.7	
5	60	11	6.6	
6	70	13	13.2	
7	70	13	9.7	
8	80	15	11.3	
9	80	15	15.6	
10				
11				

Data Science: Python Pandas DataFrame from Excel

STEPS:

1. import the pandas library
2. Create a variable for the data frame to store all columns and values from the Excel worksheet
3. Use the API from pandas pandas.read_excel(“filename.xlsx”) to read the file “filename.xlsx” and assign all columns and values to the data frame variable

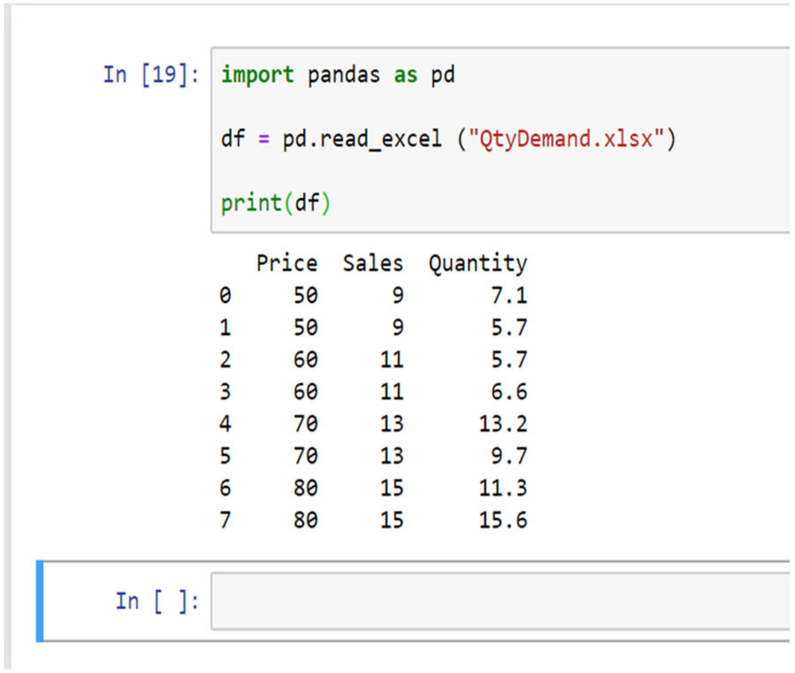
Data Science: Python Pandas DataFrame from Excel



```
In [19]: import pandas as pd

         df = pd.read_excel ("QtyDemand.xlsx")

         print(df)

            Price  Sales  Quantity
         0     50      9       7.1
         1     50      9       5.7
         2     60     11       5.7
         3     60     11       6.6
         4     70     13      13.2
         5     70     13       9.7
         6     80     15      11.3
         7     80     15      15.6

In [ ]:
```

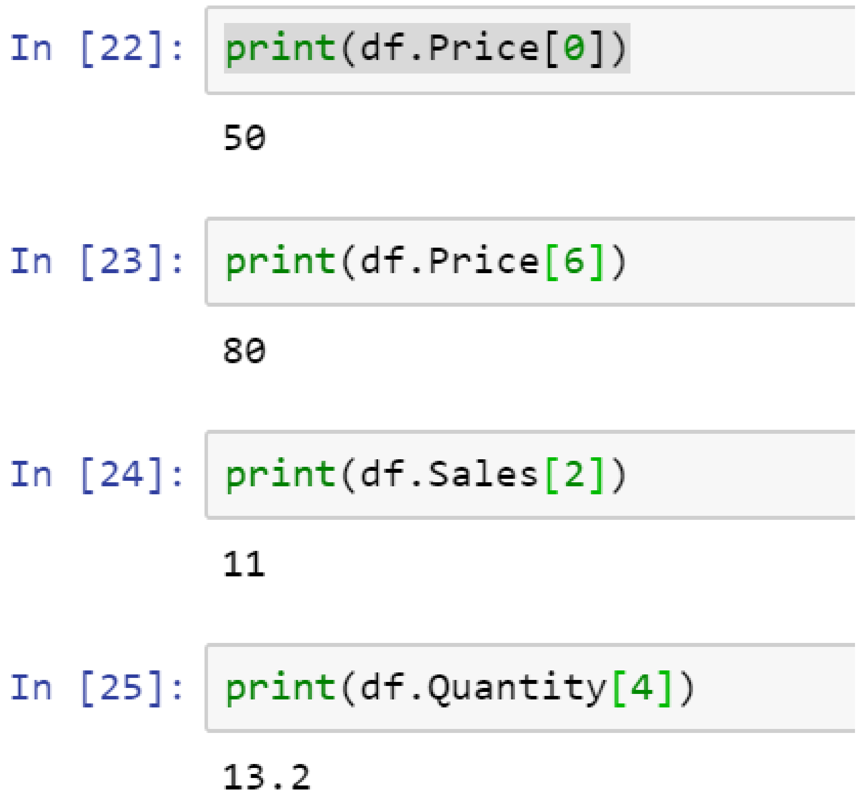
Data Science: Python Pandas DataFrame from Excel

- You can access the specific column by referencing the index (column label) of that column
  - **df[“Price”]** will return the values for the entire column “Price”
  - You can also use the form df.Price
- 0, 1, 2, 3, 4, 5, 6, 7 on the left most column is the index for the rows. You can access the value stored in column Price row 0 using **df.Price[index]**
  - **df.Price[0]** will return the value of the first element of column “Price” = 50
  - **df.Price[6]** will return the value of the seventh element of column “Price” = 80
  - **df.Sales[2]** = 11
  - **df.Quantity[4]** = 13.2

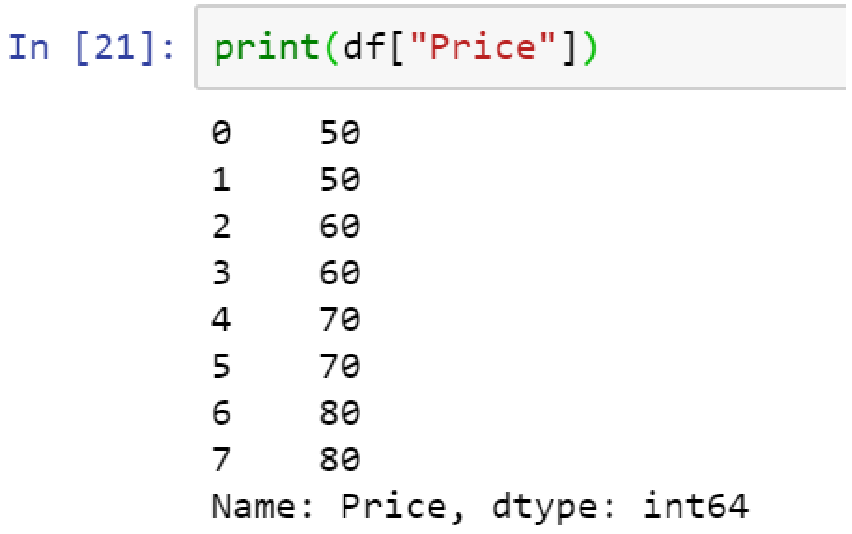


```
In [19]: import pandas as pd

         df = pd.read_excel ("QtyDemand.xlsx")

         print(df)

   Price  Sales  Quantity
0     50      9       7.1
1     50      9       5.7
2     60     11       5.7
3     60     11       6.6
4     70     13      13.2
5     70     13       9.7
6     80     15      11.3
7     80     15      15.6

In [ ]:
```

Data Science: Python Pandas DataFrame from Excel

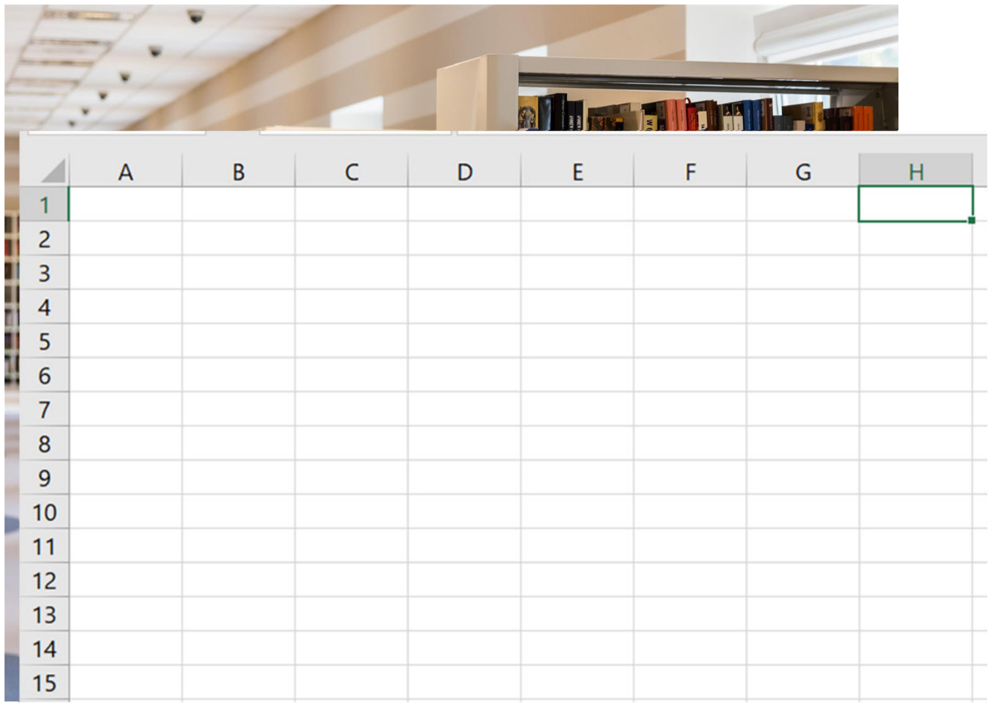


```
In [22]: print(df.Price[0])

         50

In [23]: print(df.Price[6])

         80

In [24]: print(df.Sales[2])

         11

In [25]: print(df.Quantity[4])

         13.2
```



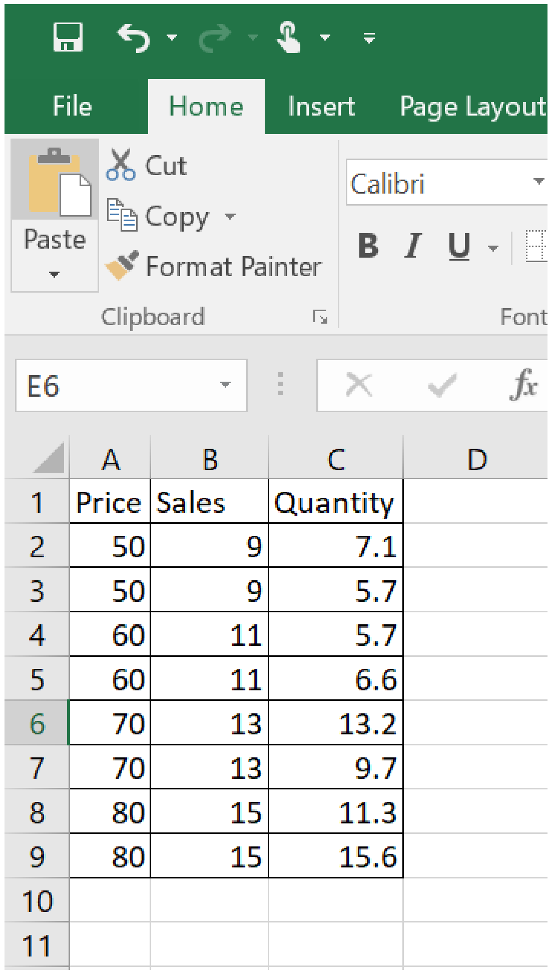
```
In [21]: print(df["Price"])

         0     50
         1     50
         2     60
         3     60
         4     70
         5     70
         6     80
         7     80
         Name: Price, dtype: int64
```

Data Science: Python Pandas DataFrame from Excel Columns



```
pandas.read_excel("filename.xlsx")
```

Data Science: Python Pandas DataFrame from Excel



	A	B	C	D
1	Price	Sales	Quantity	
2	50	9	7.1	
3	50	9	5.7	
4	60	11	5.7	
5	60	11	6.6	
6	70	13	13.2	
7	70	13	9.7	
8	80	15	11.3	
9	80	15	15.6	
10				
11				

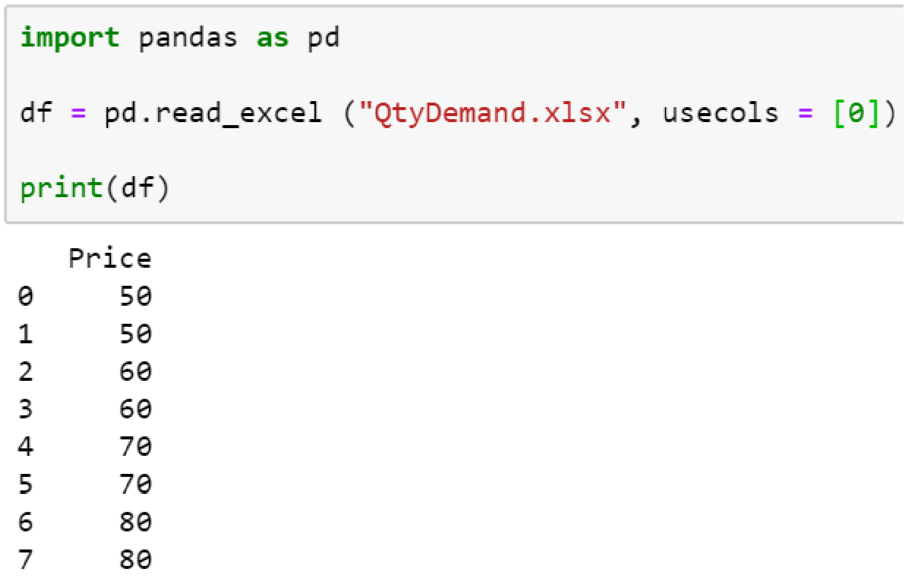
What if we only want to read column "Price" into the dataframe variable?

Data Science: Python Pandas DataFrame from Excel

STEPS:

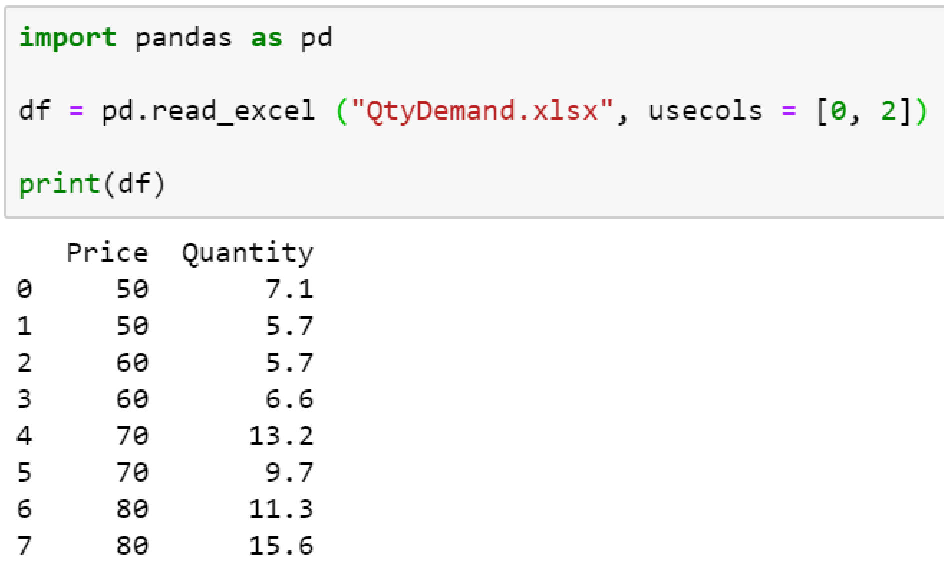
1. import the pandas library
2. Create a variable for the data frame to store all columns and values from the Excel worksheet
3. Use the API from pandas pandas.read_excel(“filename.xlsx”) to read the file “filename.xlsx” and assign all columns and values to the data frame variable.
    1. This time we will use an additional argument named usecols.
    2. pandas.read_excel(“filename.xlsx”, usecols=[0]) to read first column only
    3. pandas.read_excel(“filename.xlsx”, usecols=[0, 1]) to read first column and second column only

Data Science: Python Pandas DataFrame from Excel



```python
import pandas as pd

df = pd.read_excel ("QtyDemand.xlsx", usecols = [0])

print(df)
```

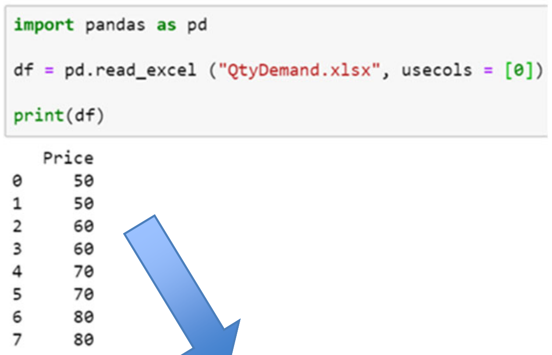
```
   Price
0     50
1     50
2     60
3     60
4     70
5     70
6     80
7     80
```



```python
import pandas as pd

df = pd.read_excel ("QtyDemand.xlsx", usecols = [0, 2])

print(df)
```

```
   Price  Quantity
0     50       7.1
1     50       5.7
2     60       5.7
3     60       6.6
4     70      13.2
5     70       9.7
6     80      11.3
7     80      15.6
```

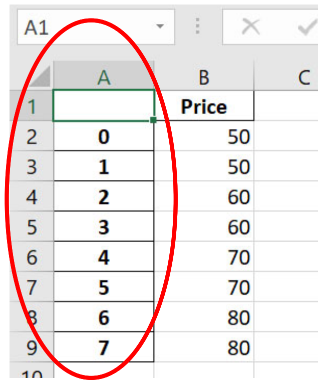
Data Science: Python Pandas DataFrame Saving to Excel File



```
import pandas as pd

df = pd.read_excel ("QtyDemand.xlsx", usecols = [0])

print(df)
```

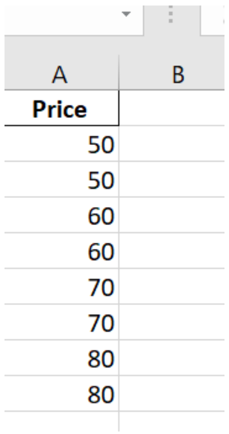
```
   Price
0     50
1     50
2     60
3     60
4     70
5     70
6     80
7     80
```

```
df.to_excel ("test2.xlsx")
```



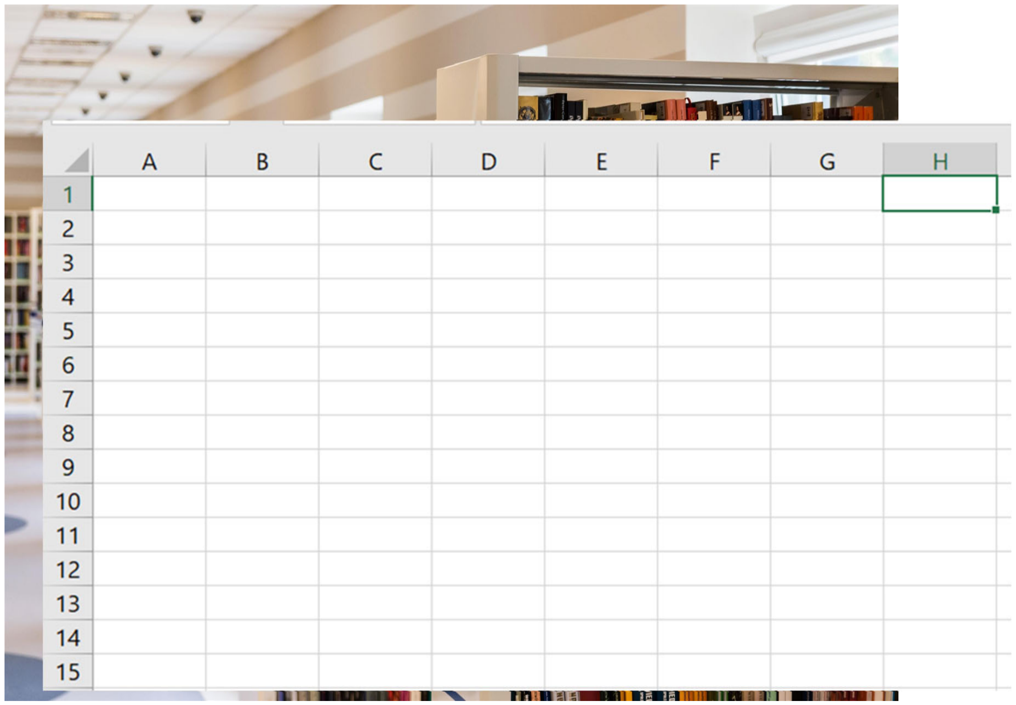
	A	B	C
1		Price	
2	0	50	
3	1	50	
4	2	60	
5	3	60	
6	4	70	
7	5	70	
8	6	80	
9	7	80	

```
df.to_excel ("test2.xlsx", index = False)
```

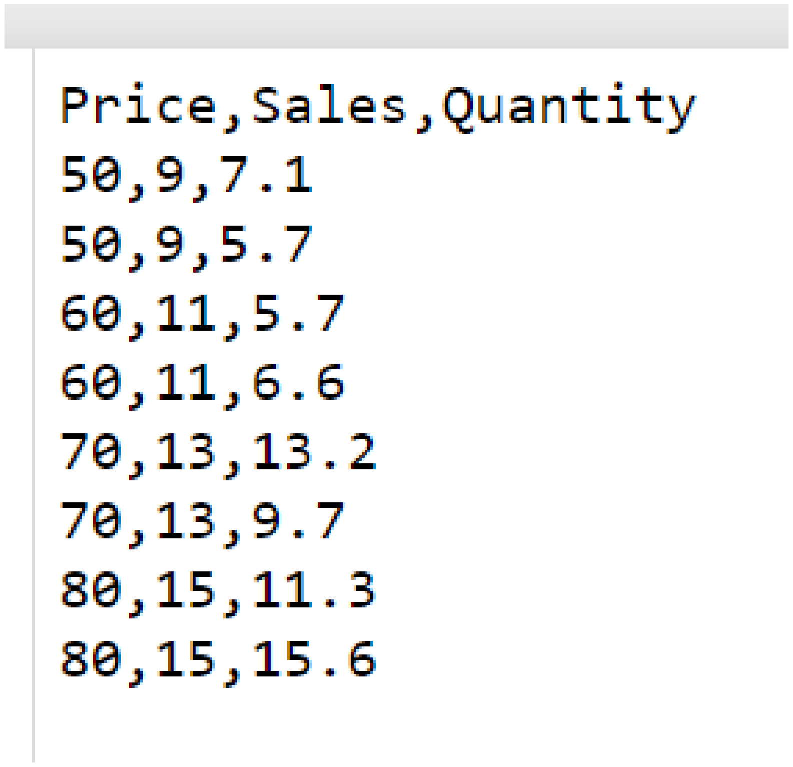


A	B
Price	
50	
50	
60	
60	
70	
70	
80	
80	

Data Science: Python Pandas DataFrame from csv



Data Science: Python Pandas DataFrame from csv



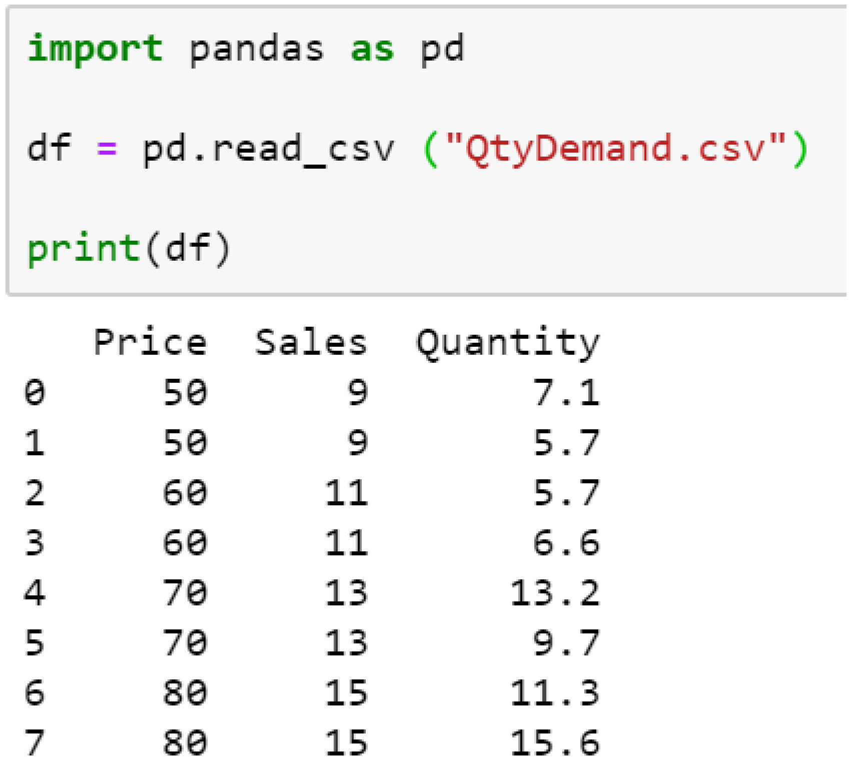
```
Price,Sales,Quantity
50,9,7.1
50,9,5.7
60,11,5.7
60,11,6.6
70,13,13.2
70,13,9.7
80,15,11.3
80,15,15.6
```

Data Science: Python Pandas DataFrame from csv

STEPS:

1. import the pandas library
2. Create a variable for the data frame to store all columns and values from the csv file
3. Use the API from pandas pandas.read_csv("filename.csv") to read the file "filename.csv" and assign all data to the data frame variable

Data Science: Python Pandas DataFrame from csv

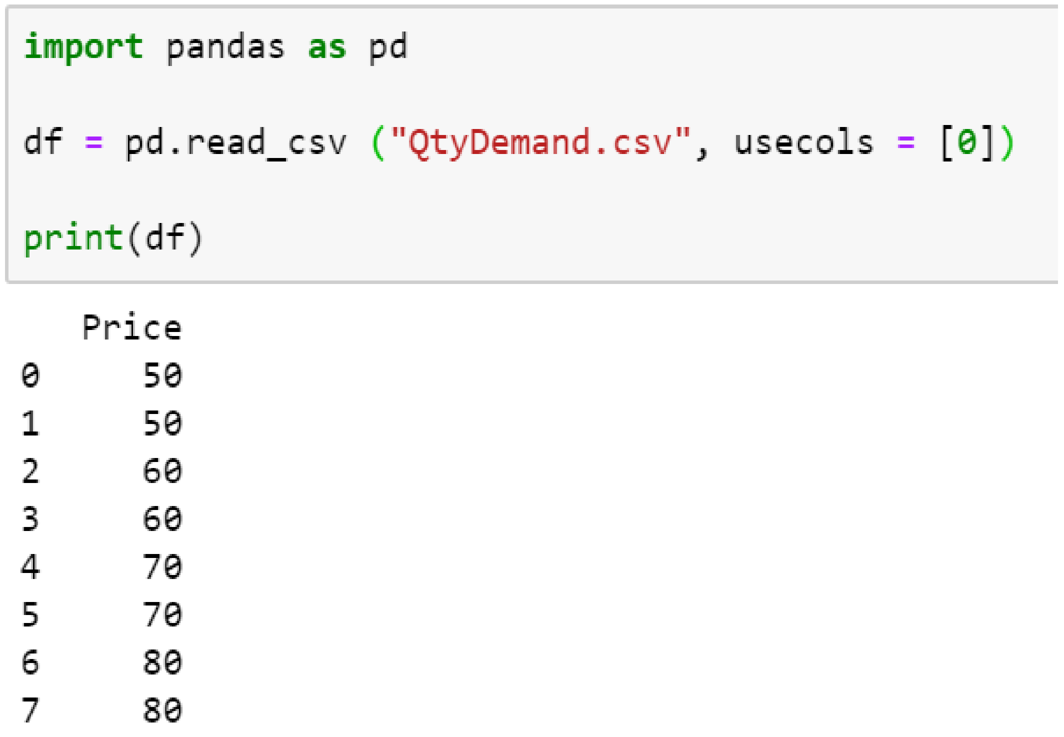


```python
import pandas as pd

df = pd.read_csv ("QtyDemand.csv")

print(df)
```

```
   Price  Sales  Quantity
0     50      9       7.1
1     50      9       5.7
2     60     11       5.7
3     60     11       6.6
4     70     13      13.2
5     70     13       9.7
6     80     15      11.3
7     80     15      15.6
```

Data Science: Python Pandas DataFrame from csv

Specific Column



```python
import pandas as pd

df = pd.read_csv ("QtyDemand.csv", usecols = [0])

print(df)
```

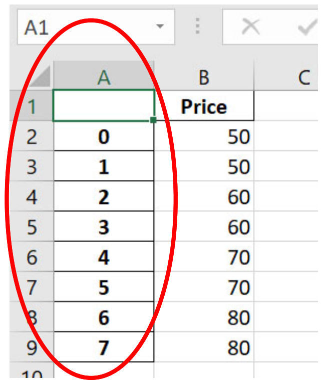
```
   Price
0     50
1     50
2     60
3     60
4     70
5     70
6     80
7     80
```

Data Science: Python Pandas DataFrame Saving to a csv file

```python
import pandas as pd

df = pd.read_csv ("QtyDemand.csv", usecols = [0])

print(df)
```

```
   Price
0     50
1     50
2     60
3     60
4     70
5     70
6     80
7     80
```

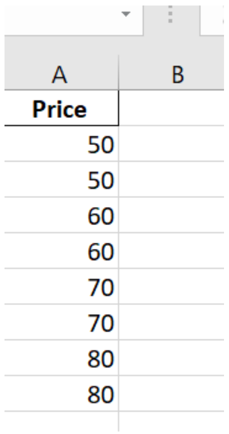
```python
df.to_csv ("test3.csv")
```



A1

	A	B	C
1		Price	
2	0	50	
3	1	50	
4	2	60	
5	3	60	
6	4	70	
7	5	70	
8	6	80	
9	7	80	

```python
df.to_csv ("test3.csv", index = False)
```



A	B
Price	
50	
50	
60	
60	
70	
70	
80	
80	